

REMARKS

Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Coker, *et al.* (U.S. Publication No. 2003/0149837) (hereafter ‘Coker’) in view of Zimmerman, *et al.* (U.S. Publication No. 2003/0200290) (hereafter ‘Zimmerman’). As will be shown below, neither Coker nor Zimmerman, either alone or in combination, teaches or suggests a method, system, or computer program product for loading data from disk in a data processing system as claimed in the present application. Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 are therefore patentable and should be allowed. Applicants respectfully traverse each rejection individually and request reconsideration of claims 1, 3, 6, 8, 10, 13-15, 17, and 20.

Claims 2, 9, and 16 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Hung (U.S. Patent No. 5,247,653) (hereafter ‘Hung’). As will be shown below, neither Coker, Zimmerman, nor Hung, either alone or in combination, teaches or suggests a method, system, or computer program product for loading data from disk in a data processing system as claimed in the present application. Claims 2, 9, and 16 are therefore patentable and should be allowed. Applicants respectfully traverse each rejection individually and request reconsideration of claims 2, 9, and 16.

Claims 4-5, 12, and 19 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Lee (U.S. Publication No. 2004/0260909) (hereafter ‘Lee’). As will be shown below, neither Coker, Zimmerman, nor Lee, either alone or in combination, teaches or suggests a method, system, or computer program product for loading data from disk in a data processing system as claimed in the present application. Claims 4-5, 12, and 19 are therefore patentable and should be allowed. Applicants respectfully traverse each rejection individually and request reconsideration of claims 4-5, 12, and 19.

Claims 7, 11, and 18 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Brady (U.S. Patent No. 5,758,050) (hereafter ‘Brady’). As will be shown below, neither Coker, Zimmerman, nor Brady, either alone or in combination, teaches or suggests a method, system, or computer program product for loading data from disk in a data processing system as claimed in the present application. Claims 7, 11, and 18 are therefore patentable and should be allowed. Applicants respectfully traverse each rejection individually and request reconsideration of claims 7, 11, and 18.

Claim Rejections – 35 U.S.C. §103

Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Coker, *et al.* (U.S. Publication No. 2003/0149837) (hereafter ‘Coker’) in view of Zimmerman, *et al.* (U.S. Publication No. 2003/0200290) (hereafter ‘Zimmerman’). Applicants respectfully traverse each rejection. To establish a prima facie case of obviousness under 35 U.S.C. § 103 the proposed combination of the references must teach or suggest all of Applicants’ claim limitations. *In re Royka*, 490 F.2d 981, 985, 180 USPQ 580, 583 (CCPA 1974). As will be shown below in more detail, the proposed combination of Coker and Zimmerman cannot establish a prima facie case of obviousness because the proposed combination does not teach each and every element of the claims of the present application. Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 are therefore patentable and should be allowed. Applicants respectfully request reconsideration of claims 1, 3, 6, 8, 10, 13-15, 17, and 20.

Independent claim 1 recites:

1. A method of loading data from disk in a data processing system, comprising:

comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests;

responsive to detecting a match between the current sequence and the previous sequence, storing a copy of data blocks accessed during the current sequence in a contiguous portion of the disk; and

responsive to a subsequent request for data in the disk sequence, mapping the request to the sequential portion of the disk and servicing the request from data in the sequential portion.

**Coker Neither Discloses Nor Suggests Comparing
A Current Sequence Of Disk Requests To Data Indicative Of
A Previous Sequence Of Disk Requests Or Detecting A Match
Between The Current Sequence And The Previous Sequence**

The Office Action takes the position that Coker at Figure 5, and paragraphs 0046, 0067, and 0068, discloses the first element of claim 1 and a portion of the second element of claim 1: comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests and detecting a match between the current sequence and the previous sequence. Applicants respectfully note in response, however, that what Coker at paragraph 0046, in fact discloses is:

[0046] The MODE DETECT subroutine of FIG. 5 generally operates to evaluate current data access patterns of the host device and dynamically configure the disc drive 100 accordingly to optimize data transfer performance. At step 216 the most recently received (latest) read command is compared to a history of recent read commands. This is preferably carried out using a read command history table as shown in FIG. 6.

And what Coker at paragraphs 0067 and 0068, in fact discloses is:

[0067] Referring again to decision step 208 of FIG. 4, it will now be assumed that a read command B (READ CMD B) 264 of FIG. 6 represents the most recently received read command. As can be observed in FIG. 6, the read command B 264 overlaps a data range block 266 associated with read command 13.

[0068] In such case, the flow of FIG. 5 passes to step 268 wherein the history table is updated with new data associated with the read command B. However, this time the new data is not used to replace the oldest entry in the table; rather, the new data preferably replaces the overlapped data block 266.

That is, Coker at paragraphs 0046, 0067, and 0068, discloses comparing the most recently received read command to a history of recent read commands and Coker at Figure 5, discloses a subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data transfer performance. Neither Coker's comparing the most recently received read command to a history of recent read commands nor Coker's subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data transfer performance discloses or suggests comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests and detecting a match between the current sequence and the previous sequence as claimed in the present application.

Neither Coker's most recently received read command compared to a history of recent read commands nor Coker's subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data transfer performance discloses or suggests comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests as claimed in the present application. Coker only discloses comparing a single read command – not a sequence of disk requests as claimed in the present application - to a history of read commands. That is, Coker does not compare a sequence of any requests to data indicative of a previous sequence of disk requests as claimed in the present application. Coker does not disclose or suggest therefore comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests and detecting a match between the current sequence and the previous sequence.

Neither Coker's comparing the most recently received read command to a history of recent read commands nor Coker's subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data

transfer performance discloses or suggests detecting a match between the current sequence and the previous sequence as claimed in the present application. A match as claimed in the present application is detected by comparing *a current sequence* of disk requests to data indicative of a previous sequence of disk requests. Coker, however, does not disclose or suggest comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests as claimed in the present application. Because Coker does not disclose or suggest comparing a current sequence of disk requests to data indicative of a previous sequence of disk requests as claimed in the present application Coker cannot disclose detecting a match between the current sequence and the previous sequence as claimed in the present application.

In addition to the fact that nether Coker's comparing the most recently received read command to a history of recent read commands nor Coker's subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data transfer performance discloses or suggests a match between the current sequence and the previous sequence as claimed in the present application, there is a second reason that Coker does not disclose detecting a match between the current sequence and the previous sequence as claimed in the present application: Detecting a match between the current sequence of disk requests and the previous sequence of disk requests as claimed in the present application occurs so that a copy of data blocks accessed during the current sequence can be stored in a contiguous portion of the disk. Coker's 'subroutine' only updates a history table with new data associated with a read command when the read command overlaps a data range block associated with a previous read command. Updating a history table does not disclose storing a copy of data blocks in a contiguous portion of the disk as claimed in the present application. In fact, Coker does not disclose, at these reference points or anywhere else, a 'contiguous portion' as claimed in the present application. Therefore, neither Coker's comparing the most recently received read command to a history of recent read commands nor Coker's subroutine that operates to evaluate current data access patterns and dynamically configure the disc drive accordingly to optimize data transfer performance discloses or suggests detecting a match between the current sequence and the previous sequence as

claimed in the present application. The Office Action therefore cannot establish a prima facie case of obviousness. The rejections of claims 1, 3, 6, 8, 10, 13-15, 17, and 20 should be withdrawn, and the claims should be allowed.

**Zimmerman Neither Discloses Nor Suggests Storing
A Copy Of Data Blocks Accessed During The Current Sequence
In A Contiguous Portion Of The Disk Or Responsive To A Subsequent
Request For Data In The Disk Sequence, Mapping The Request
To The Sequential Portion Of The Disk And Servicing The
Request From Data In The Sequential Portion**

The Office Action takes the position that Zimmerman at paragraph 0042, and Figure 4, discloses the following portion of the second element of claim 1 and the third element of claim 1: storing a copy of data blocks accessed during the current sequence in a contiguous portion of the disk and responsive to a subsequent request for data in the disk sequence, mapping the request to the sequential portion of the disk and servicing the request from data in the sequential portion. Applicants respectfully note in response, however, that what Zimmerman at paragraph 42, in fact discloses is:

[0042] The learning process 450, is executed if the streaming module 26 cannot find the sector sequence file 22. In one embodiment, the sector sequence file is comprised of a list of sectors that the O/S must read in order to complete the transmission of the disk image transmission. In another embodiment, the sector sequence file is comprised not only the list of sectors but also the sequentially stored actual data contained in the listed sectors. In yet another embodiment, the sector sequence file is comprised a single file including a plurality sector lists and corresponding sector data, such that the simultaneous streaming of different data sets to different sets of requesting clients may be supported. A benefit of storing the actual data is that reading a sequential file is much faster than random reads, and it will in turn increase the server's drive throughput and the ability to more efficiently support multiple streams to multiple clients. One result is that the learned actual data can be read exclusively from the sector sequence file until all of the learned data has been transferred to the client. After that point, the server can revert back to using the virtual drive image. In step 426, the streaming module 26 selects one client from the set of registered clients 2'. In step 428, the selected client is permitted to make its disk access requests conventionally, while the streaming module records in a new sector sequence file all sector access requests the selected

client makes to fulfill its desired data download, and, in certain embodiments, the requested data itself. In step 430, the selected client informs the streaming module 26 that it has completed its download. At this point, the new sector sequence file is stored on the virtual drive 8, and the streaming process is resumed at step 408. Described below is a use of the streaming process in network booting applications. If the learning process 450 is required to create a sector sequence file 22 in the context of network booting, the selected client is permitted to boot conventionally, using the inventive MBR 24 and drivers (30 and 32), while the streaming module 26 records in a new sector sequence file all sector access requests the selected client makes while booting, and, in some embodiments, the actual data requested.

That is, Zimmerman at paragraph 0042 and Figure 4, discloses recording in a new sector sequence file all sector access requests a client makes and the requested data.

Zimmerman's recording in a new sector sequence file all sector access requests a client makes and the requested data does not disclose or suggest storing a copy of data blocks accessed during the current sequence in a contiguous portion of the disk and responsive to a subsequent request for data in the disk sequence, mapping the request to the sequential portion of the disk and servicing the request from data in the sequential portion as claimed in the present application. Storing a copy of data blocks accessed during the current sequence as claimed in the present application occurs in response to detecting a match between the current sequence and the previous sequence of disk requests.

Zimmerman does not disclose or suggest, at these reference points or any other where else, detecting a match between a current sequence and previous sequence of disk requests as claimed in the present application. Because Zimmerman does not disclose or suggest detecting a match between the current sequence and the previous sequence of disk requests as claimed in the present application Zimmerman cannot disclose or suggest storing, in response to detecting such a match, a copy of data blocks accessed during the current sequence in a contiguous portion of the disk as claimed in the present application.

Zimmerman does not disclose or suggest responsive to a subsequent request for data in the disk sequence, mapping the request to the sequential portion of the disk and servicing the request from data in the sequential portion as claimed in the present application.

Zimmerman does not disclose or suggest storing a copy of data blocks accessed during

the current sequence in the sequential portion of the disk and as such cannot disclose or suggest mapping any request to such sequential portion of the disk as claimed in the present application. Zimmerman does not disclose or suggest, therefore, responsive to a subsequent request for data in the disk sequence, mapping the request to the sequential portion of the disk and servicing the request from data in the sequential portion as claimed in the present application. The Office Action therefore cannot establish a prima facie case of obviousness. The rejections of claims 1, 3, 6, 8, 10, 13-15, 17, and 20 should be withdrawn, and the claims should be allowed.

**Claim Rejections – 35 U.S.C. § 103 Over The Combination Of
Coker And Zimmerman In View of Hung**

Claims 2, 9, and 16 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Hung (U.S. Patent No. 5,247,653) (hereafter ‘Hung’). To establish a prima facie case of obviousness, the proposed combination of the references must teach or suggest all of the claim limitations of dependent claims 2, 9, and 16. *In re Royka*, 490 F.2d 981, 985, 180 USPQ 580, 583 (CCPA 1974). Dependent claims 2, 9, and 16 depend from independent claims 1, 8, and 14 and include all the limitations of the independent claims from which they depend. In rejecting dependent claims 2, 9, and 16, the Office Action relies on the combination of Coker and Zimmerman as disclosing each and every element of independent claims 1, 8, and 14. As shown above, the combination of Coker and Zimmerman in fact does not disclose each and every element of independent claims 1, 8, and 14. Because the combination of Coker and Zimmerman does not disclose each and every element of independent claims 1, 8, and 14, the combination of the combination of Coker, Zimmerman, and Hung cannot possibly disclose each and every element of dependent claims 2, 9, and 16. The proposed combination Coker, Zimmerman, and Hung, therefore, cannot establish a prima facie case of obviousness, and the rejections 35 U.S.C. § 103(a) should be withdrawn.

**Claim Rejections – 35 U.S.C. § 103 Over The Combination Of
Coker And Zimmerman In View of Lee**

Claims 4-5, 12, and 19 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Lee (U.S. Publication No. 2004/0260909) (hereafter ‘Lee’). To establish a prima facie case of obviousness, the proposed combination of the references must teach or suggest all of the claim limitations of dependent claims 4-5, 12, and 19. *In re Royka*, 490 F.2d 981, 985, 180 USPQ 580, 583 (CCPA 1974). Dependent claims 4-5, 12, and 19 depend from independent claims 1, 8, and 14 and include all the limitations of the independent claims from which they depend. In rejecting dependent claims 4-5, 12, and 19, the Office Action relies on the combination of Coker and Zimmerman as disclosing each and every element of independent claims 1, 8, and 14. As shown above, the combination of Coker and Zimmerman in fact does not disclose each and every element of independent claims 1, 8, and 14. Because the combination of Coker and Zimmerman does not disclose each and every element of independent claims 1, 8, and 14, the combination of the combination of Coker, Zimmerman, and Lee cannot possibly disclose each and every element of dependent claims 4-5, 12, and 19. The proposed combination Coker, Zimmerman, and Lee, therefore, cannot establish a prima facie case of obviousness, and the rejections 35 U.S.C. § 103(a) should be withdrawn.

**Claim Rejections – 35 U.S.C. § 103 Over The Combination Of
Coker And Zimmerman In View of Brady**

Claims 7, 11, and 18 stand rejected for obviousness under 35 U.S.C. § 103(a) as being unpatentable over the combination of Coker and Zimmerman in view of Brady (U.S. Patent No. 5,758,050) (hereafter ‘Brady’). To establish a prima facie case of obviousness, the proposed combination of the references must teach or suggest all of the claim limitations of dependent claims 7, 11, and 18. *In re Royka*, 490 F.2d 981, 985, 180 USPQ 580, 583 (CCPA 1974). Dependent claims 7, 11, and 18 depend from independent claims 1, 8, and 14 and include all the limitations of the independent claims from which they depend. In rejecting dependent claims 7, 11, and 18, the Office Action relies on the

combination of Coker and Zimmerman as disclosing each and every element of independent claims 1, 8, and 14. As shown above, the combination of Coker and Zimmerman in fact does not disclose each and every element of independent claims 1, 8, and 14. Because the combination of Coker and Zimmerman does not disclose each and every element of independent claims 1, 8, and 14, the combination of the combination of Coker, Zimmerman, and Brady cannot possibly disclose each and every element of dependent claims 7, 11, and 18. The proposed combination Coker, Zimmerman, and Brady, therefore, cannot establish a prima facie case of obviousness, and the rejections 35 U.S.C. § 103(a) should be withdrawn.

Relations Among Claims

Independent claim 1 claims method aspects of loading data from disk in a data processing system according to embodiments of the present invention. Independent claims 8 and 14 respectively claim computer program product and system aspects of loading data from disk in a data processing system according to embodiments of the present invention. Claim 1 is allowable for the reasons set forth above. Claims 8 and 14 are allowable because claim 1 is allowable. The rejections of claims 8 and 14 therefore should be withdrawn, and claims 8 and 14 should be allowed.

Claims 2-7, 9-13, and 15-20 depend respectively from independent claims 1, 8, and 14. Each dependent claim includes all of the limitations of the independent claim from which it depends. Because the combination of Coker and Zimmerman does not disclose or suggest each and every element of the independent claims, so also the combination of Coker and Zimmerman cannot possibly disclose or suggest each and every element of any dependent claim. The rejections of Claims 2-7, 9-13, and 15-20 therefore should be withdrawn, and these claims also should be allowed.

Conclusion

Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 stand rejected under 35 U.S.C. § 103 as obvious over Coker in view of Zimmerman. The combination of Coker and Zimmerman does not teach or suggest each and every element of Applicants' claims. Claims 1, 3, 6, 8, 10, 13-15, 17, and 20 are therefore patentable and should be allowed. Applicants respectfully request reconsideration of claims 1, 3, 6, 8, 10, 13-15, 17, and 20.

Claims 2, 9, and 16 stand rejected under 35 U.S.C. § 103 as obvious over the combination of Coker and Zimmerman in view of Hung. The combination of Coker, Zimmerman, and Hung does not teach or suggest each and every element of Applicants' claims. Claims 2, 9, and 16 are therefore patentable and should be allowed. Applicants respectfully request reconsideration of claims 2, 9, and 16.

Claims 4-5, 12, and 19 stand rejected under 35 U.S.C. § 103 as obvious over the combination of Coker and Zimmerman in view of Lee. The combination of Coker, Zimmerman, and Lee does not teach or suggest each and every element of Applicants' claims. Claims 4-5, 12, and 19 are therefore patentable and should be allowed. Applicants respectfully request reconsideration of claims 4-5, 12, and 19.

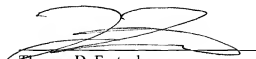
Claims 7, 11, and 18 stand rejected under 35 U.S.C. § 103 as obvious over the combination of Coker and Zimmerman in view of Brady. The combination of Coker, Zimmerman, and Brady does not teach or suggest each and every element of Applicants' claims. Claims 7, 11, and 18 are therefore patentable and should be allowed. Applicants respectfully request reconsideration of claims 7, 11, and 18.

The Commissioner is hereby authorized to charge or credit Deposit Account No. 50-0563 for any fees required or overpaid.

Respectfully submitted,

Date: April 17, 2007

By:

A handwritten signature in black ink, appearing to read 'Thomas D. Fortenberry', written over a horizontal line.

Thomas D. Fortenberry
Reg. No. 56,537
Biggers & Ohanian, LLP
P.O. Box 1469
Austin, Texas 78767-1469
Tel. (512) 472-9881
Fax (512) 472-9887
ATTORNEY FOR APPLICANTS